

=====

Tutor: **Rahul Shetty**

Reference: **UDEMY**

Course: **Cypress - Modern Automation Testing from Scratch + Frameworks**

Content: **Programming language - JAVASCRIPT**

=====

1. Course URL: <https://www.udemy.com/course/cypress-tutorial/>
2. Document prepared by: **Rajat Verma**
 - a. <https://www.linkedin.com/in/rajat-v-3b0685128/>
 - b. <https://github.com/rajatt95>
 - c. <https://rajatt95.github.io/>

Softwares:

1. Programming language - Javascript
2. Node JS
3. IDE - Visual Studio Code

1. Learnings from Course (UDEMY - RS - Cypress)

a. Links:

- i. <https://nodejs.org/en/download/>
- ii. <https://code.visualstudio.com/download>
- iii. <https://www.npmjs.com/>
- iv. <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

b. Javascript fundamentals for Automation Testing

- i. Variables declaration and assignment
 1. typeof()
- ii. Decision making
 1. If-Else
- iii. Loops
 1. For
 2. While
 3. Do While
- iv. Keywords
 1. var
 2. let

3. const
- v. Arrays and operations
 1. push()
 2. pop()
 3. unshift()
 4. indexOf()
 5. includes()
 6. slice()
 7. filter()
 8. map()
 9. sort()
 10. reverse()
- vi. Functions
 1. Custom
 2. Anonymous
- vii. String
 1. length
 2. charAt()
 3. slice()
 4. indexOf()
 5. split()
 6. trim()
 7. parseInt()
 8. toString()
- viii. Javascript Object
 1. Properties
 - a. Single value
 - b. As Anonymous function
- ix. Classes and Objects
 1. Same clas
 2. Different class
 - a. Export the class
 - b. Import it and create the object of that class
- x. OOPS
 1. Inheritance

=====

=====21_Learn JavaScript Fundamentals from Scratch for
Automation=====

1. Javascript fundamentals for Automation Testing

- a. File Extension - **.js**
- b. How to run/execute the program - **node 01_HelloWorld.js**
 - i. 01_HelloWorld.js is the file name

2. Programs:

a. Hello World:

```
JS 01_HelloWorld.js ×  
  
JS 01_HelloWorld.js  
1 // This is single line comment  
2  
3 /*  
4 This is Multi-line comment  
5 */  
6  
7 //Java  
8 //System.out.println("Hello, Test Automation Engineer!");  
9  
10 //Javascript  
11 console.log("Hello, Test Automation Engineer!")
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 01_HelloWorld.js  
Hello, Test Automation Engineer!  
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.

b. Declaring variable:

- i. Variables in Javascript are loosely coupled
 1. Variables can hold the value of any type of data
 2. Variables can define the nature of value at runtime
- ii. **var** is a keyword in Javascript
- iii. From ES6 engine,
 1. Two more keywords came in the picture
 - a. let
 - b. const

```
JS 02_DeclaringVariables.js ×
JS 02_DeclaringVariables.js > [?] str1
1 // This is single line comment
2
3 /*
4 This is Multi-line comment
5 */
6
7 //Java
8 //int num1=10;
9 //String str1=19;
10 //System.out.println("num: "+num1);
11 //System.out.println("str: "+str1);
12
13
14 //Javascript
15 var num1=10
16 var str1="Test Automation Engineer"
17 myFloat=198.124
18
19 console.log("num: "+num1)
20 console.log("str: "+str1)
21 console.log("myFloat: "+myFloat)
```

iv.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 02_DeclaringVariables.js
num1: 10
str1: Test Automation Engineer
myFloat: 198.124
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

v.

c. Type of

- i. There is nothing like float, double in Javascript
 - 1. 16, 16.89, 1666666666.8 -> these are numbers

```
JS 03_DeclaringVariables_TypeOf.js ×  
JS 03_DeclaringVariables_TypeOf.js > ...  
1 var num1=10  
2 let num2=113.3456  
3 var str1="Test Automation Engineer"  
4 let myBoolean=true  
5  
6 console.log("num1: "+num1)  
7 console.log("num2: "+num2)  
8 console.log("str1: "+str1)  
9 console.log("myBoolean"+myBoolean)  
10  
11  
12 console.log("typeof(num1): "+typeof(num1))  
13 console.log("typeof(num2): "+typeof(num2))  
14 console.log("typeof(str1): "+typeof(str1))  
15 console.log("typeof(myBoolean): "+typeof(myBoolean))
```

ii.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 03_DeclaringVariables_TypeOf.js  
num1: 10  
num2: 113.3456  
str1: Test Automation Engineer  
myBooleantrue  
typeof(num1): number  
typeof(num2): number  
typeof(str1): string  
typeof(myBoolean): boolean  
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

iii.

d. Mathematical Operations

```
JS 04_Variables_MathematicalOperations.js ×
JS 04_Variables_MathematicalOperations.js > ...
1
2   var num1=200
3   let num2=100.1
4
5   console.log("num1+num2: "+num1+num2)
6   console.log("num1-num2: "+num1-num2)
7   console.log("num1*num2: "+num1*num2)
8   console.log("num1/num2: "+num1/num2)
9   console.log("num1%num2: "+num1%num2)
10
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 04_Variables_MathematicalOperations.js
num1+num2: 200100.1
NaN
num1*num2: 20020
num1/num2: 1.998001998001998
num1%num2: 99.9
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.

e. Value Re-Assignment

```
JS 05_Variables_ValueReAssignment.js ×
JS 05_Variables_ValueReAssignment.js > ...
1
2   var num1=200
3   console.log("num1: "+num1)
4
5   num1=500 //Re-assigning the value to variable num1
6   console.log("num1: "+num1)
7
8   //////////////////////////////////////
9
10  let num2=100
11  console.log("num2: "+num2)
12
13  num2=300 //Re-assigning the value to variable num2
14  console.log("num2: "+num2)
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 05_Variables_ValueReAssignment.js
num1: 200
num1: 500
num2: 100
num2: 300
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.

f. If-Else

```
JS 06_IfElse.js ×
JS 06_IfElse.js > ...
1  const flag=true
2
3  if(flag){
4      console.log("Condition satisfied")
5      console.log("flag: "+flag)
6  }else{
7      console.log("Condition not satisfied")
8  }
9
10 console.log("-----")
11
12 console.log("Checking the reverse of flag")
13 if(!flag){
14     console.log("Condition satisfied")
15     console.log("flag: "+flag)
16 }else{
17     console.log("Condition not satisfied")
18 }
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 06_IfElse.js
Condition satisfied
flag: true
-----
Checking the reverse of flag
Condition not satisfied
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.

g. Loop - While

```
JS 07_Loop_While.js x
JS 07_Loop_While.js > ...
1   let num1=1;
2
3   while(num1<11){
4       console.log(num1)
5
6       //Post-increment
7       num1++
8   }
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 07_Loop_While.js
1
2
3
4
5
6
7
8
9
10
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.

h. Loop-Do While

```
JS 08_Loop_DoWhile.js x
JS 08_Loop_DoWhile.js > ...
1   let num1=1
2
3   //In Do-While Loop, 1 round will execute for sure
4   do{
5       console.log(num1)
6
7       //Post-increment
8       num1++
9   }while(num1<11)
10
11  console.log("-----")
12
13  num1=1
14  //In Do-While Loop, 1 round will execute for sure
15  do{
16      console.log(num1)
17
18      //Post-increment
19      num1++
20  }while(num1<1) //Even if the condition is false, Code inside the do section executed for one time.
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 08_Loop_DoWhile.js
1
2
3
4
5
6
7
8
9
10
-----
1
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.

i. Loop - For

```
JS 09_Loop_For.js ×  
  
JS 09_Loop_For.js > ...  
1   for (let index = 0; index < 11; index++) {  
2     console.log(index)  
3   }  
4  
5   console.log("Success")  
  
i.  
  
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 09_Loop_For.js  
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
Success  
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

j. Arrays

```
JS 10_Arrays.js ×  
  
JS 10_Arrays.js > ...  
1  
2   //1st way  
3   //let marks1 = Array(6)  
4  
5   //2nd way  
6   //let marks2 = new Array(10,20,30,40,50,60)  
7  
8   //3rd way  
9   let marks3 = [11,22,33,44,55,66]  
10  console.log("marks3: "+marks3)  
11  
12  for (let index = 0; index < marks3.length; index++) {  
13    console.log(marks3[index])  
14  }  
15  
16  console.log("Success")  
  
i.
```

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 10_Arrays.js
marks3: 11,22,33,44,55,66
11
22
33
44
55
66
Success
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.

k. Array Operations:

```
JS 11_Arrays_Operations_Push_Pop.js ×
JS 11_Arrays_Operations_Push_Pop.js > ...
4
5 //2nd way
6 //let marks2 = new Array(10,20,30,40,50,60)
7
8 //3rd way
9 let marks3 = [11,22,33,44,55,66]
10 console.log("marks3: "+marks3) //11,22,33,44,55,66
11
12 console.log("push() will add the element at last index in the Array")
13 marks3.push(77)
14 console.log("marks3: "+marks3) //11,22,33,44,55,66,77
15
16 console.log("pop() will delete the last added element in the Array")
17 marks3.pop()
18 console.log("marks3: "+marks3) //11,22,33,44,55,66
19
20 console.log("Success")
```

i.

```
JS 12_Arrays_Operations_Unshift.js ×
JS 12_Arrays_Operations_Unshift.js > ...
1 let marks3 = [11,22,33,44,55,66]
2
3 console.log("marks3: "+marks3) //11,22,33,44,55,66
4
5 console.log("unshift() will add the element at first index in the Array")
6 marks3.unshift(77)
7 console.log("marks3: "+marks3) //77,11,22,33,44,55,66
8
9 console.log("Success")
```

ii.

iii.

```
JS 13_Arrays_Operations_LocateValue.js ×
JS 13_Arrays_Operations_LocateValue.js > ...
1   let marks3 = [11,22,33,44,55,66]
2
3   console.log("marks3: "+marks3) //11,22,33,44,55,66
4
5   console.log("marks3.indexOf(33): "+marks3.indexOf(33)) //2
6
7   console.log("Success")
```

iv.

```
JS 14_Arrays_Operations_IsElementPresent.js ×
JS 14_Arrays_Operations_IsElementPresent.js > ...
1   let marks3 = [11,22,33,44,55,66]
2
3   console.log("marks3: "+marks3) //11,22,33,44,55,66
4
5   //includes() -> Will return an boolean value, will check whether the element is present in Array
6   console.log("marks3.includes(33): "+marks3.includes(33)) //true
7   console.log("marks3.includes(77): "+marks3.includes(77)) //false
8
9   console.log("Success")
```

v.

```
JS 15_Arrays_Operations_Slice.js ×
JS 15_Arrays_Operations_Slice.js > ...
1   let marks3 = [11,22,33,44,55,66]
2
3   console.log("marks3: "+marks3) //11,22,33,44,55,66
4
5   //slice() -> will slice the Array starting from index 1 to index 3
6   console.log("marks3.slice(1,4): "+marks3.slice(1,4)) //22,33,44
7
8   console.log("marks3: "+marks3) //11,22,33,44,55,66
9
10  console.log("Success")
```

```

JS 16_Arrays_Operations_Filter.js > ...
1 // Print only even
2 let marks3 = [11,22,33,44,55,66]
3
4 console.log("marks3: "+marks3) //11,22,33,44,55,66
5
6 console.log("Printing only even using For loop")
7 for (let index = 0; index < marks3.length; index++) {
8     if(marks3[index] %2 ==0){
9         console.log(marks3[index]) // 22,44,66
10    }
11 }
12 console.log("-----")
13
14 console.log("Printing only even using filter()")
15 console.log(marks3.filter(marks3 => marks3%2 ==0))
16
17 console.log("Success")

```

vi.

```

JS 17_Arrays_Operations_Map.js > ...
1 // Multiply the elements by 3
2 let marks3 = [11,22,33]
3
4 console.log("marks3: "+marks3) //11,22,33
5
6 //map() -> We are doing some mapping with all the elements present in Array
7 console.log("marks3.map(marks3 => marks3*3): "+marks3.map(marks3 => marks3*3)) //33,66,99
8
9 console.log("Success")

```

vii.

```

JS 18_Arrays_Operations_Sort.js > [🔗] stringArray
1 // Multiply the elements by 3
2 let numberArray = [11,44,22,55,33]
3
4 console.log("Before Sorting")
5 console.log("numberArray: "+numberArray) //11,44,22,55,33
6 console.log("After Sorting")
7 console.log("numberArray: "+numberArray.sort()) //11,22,33,44,55
8
9 console.log("-----")
10
11 let stringArray = ["Banana", "Orange", "Papaya", "Apple"]
12
13 console.log("Before Sorting")
14 console.log("stringArray: "+stringArray) //Banana,Orange,Papaya,Apple
15 console.log("After Sorting")
16 console.log("stringArray: "+stringArray.sort()) //Apple,Banana,Orange,Papaya
17 console.log("Success")

```

viii.

```

JS 19_Arrays_Operations_SortLogic.js > ...
1 // Multiply the elements by 3
2 let numberArray = [11,44,003,22,55,33]
3
4 console.log("numberArray: "+numberArray) //11,44,3,22,55,33
5
6 // Not working as expected
7 console.log("numberArray.sort(): "+numberArray.sort()) //11,22,33,44,55
8
9 //Working as expected
10 // Here, we are looking for minimum difference between 2 elements present in the Array
11 console.log("numberArray.sort((a,b) => a-b): "+numberArray.sort(
12 | (a,b) => a-b
13 | )) //11,22,33,44,55
14 console.log("Success")
15

```

ix.

```

JS 21_Arrays_Operations_ReverseLogic.js > ...
1 // Multiply the elements by 3
2 let numberArray = [11,44,003,22,55,33]
3
4 console.log("numberArray: "+numberArray) //11,44,3,22,55,33
5
6 // Not working as expected
7 console.log("numberArray.sort(): "+numberArray.sort()) //11,22,3,33,44,55
8
9 //Working as expected
10 // Here, we are looking for minimum difference between 2 elements present in the Array
11 console.log("numberArray.sort((a,b) => b-a): "+numberArray.sort(
12 | // (a,b) => a-b
13 | (a,b) => b-a
14 | )) //55,44,33,22,11,3
15 console.log("Success")
16

```

x.

I. Functions

```
JS 22_Function_AddTwoNumbers.js > ...
1
2   function add(a,b){
3     sum=a+b
4     console.log(a+"+"+b+" = "+sum)
5     return sum
6   }
7
8   add(2,3)
9   add(20,30)
10  add(26,34)
11  add(21,32)
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 22_Function_AddTwoNumbers.js
2+3 = 5
20+30 = 50
26+34 = 60
21+32 = 53
```

ii.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

m. Anonymous function

```
JS 23_Function_Anonymous_AddTwoNumbers.js > ...
1
2   //Anonymous function -> Function without name
3   //Anonymous function can only be assigned to variables
4
5   let sumOfTwoNumbers = function(a,b){
6     sum=a+b
7     console.log(a+"+"+b+" = "+sum)
8     return sum
9   }
10
11  sumOfTwoNumbers(10,20) //10+20 = 30
12
13  console.log("-----")
14
15  let sumOfTwo2Numbers = (a,b) => console.log(a+"+"+b+" = "+(a+b))
16  sumOfTwo2Numbers(20,50)
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 23_Function_Anonymous_AddTwoNumbers.js
10+20 = 30
-----
```

ii.

```
20+50 = 70
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

n. String

```
JS 24_String_Operations.js > [?] stringValueWithSpace
1
2 //Both Single cort and double corts
3 let stringValue='Test Automation Engineer'
4
5 console.log("stringValue.length: "+stringValue.length) //24
6
7 console.log("stringValue.charAt(0): "+stringValue.charAt(0)) //T
8 console.log("stringValue.charAt(2): "+stringValue.charAt(2)) //s
9
10 console.log("stringValue.slice(0,4): "+stringValue.slice(0,4)) //Test
11
12 console.log("indexOf() - START")
13 console.log("stringValue.indexOf('A'): "+stringValue.indexOf('A')) //5
14 console.log("stringValue.indexOf('e'): "+stringValue.indexOf('e')) //1
15 //e is present 3 times in the given String -> I want to start search from index 4, then
16 console.log("stringValue.indexOf('e',4): "+stringValue.indexOf('e',4)) //21
17 console.log("indexOf() - END")
18
19
20 console.log("stringValue[2]: "+stringValue[2]) //s
21
22 console.log("stringValue.split("\\ \\"): "+stringValue.split(" ")) //Test,Automation,Engineer
23
24 console.log("trim() - START")
25 let stringValueWithSpace = ' Hi Bye '
26 console.log("stringValueWithSpace.length: "+stringValueWithSpace.length) //8
27 console.log("stringValueWithSpace.trim().length: "+stringValueWithSpace.trim().length) //6
28 console.log("trim() - END")
```

i.

```
JS 25_String_ConvertStringToNumber.js > ...
1
2 let startDate='23'
3 let endDate='27'
4
5 console.log("endDate-startDate: "+endDate-startDate) //NaN
6
7 console.log("parseInt(startDate): "+parseInt(startDate))
8 console.log("parseInt(endDate): "+parseInt(endDate))
9
10 let diffDate = parseInt(endDate) - parseInt(startDate)
11 console.log("parseInt(endDate) - parseInt(startDate): "+diffDate)
```

ii.

o. Javascript Object

```
JS 27_JavascriptObject.js > ...
1 //Object -> It is a collection of properties
2
3 let person ={
4     firstName: 'Rajat',
5     lastName: 'Verma',
6     country: 'India'
7 }
8
9 console.log("1st way")
10 console.log("person.firstName: "+person.firstName)
11 console.log("person.lastName: "+person.lastName)
12 console.log("person.country: "+person.country)
13
14 console.log("2nd way")
15 console.log("person[firstName]: "+person['firstName'])
16 console.log("person[lastName]: "+person['lastName'])
17 console.log("person[country]: "+person['country'])
18
```

i.

```
rajatverma@rajats-MacBook-Air _01_JavascriptBasics % node 27_JavascriptObject.js
1st way
person.firstName: Rajat
person.lastName: Verma
person.country: India
2nd way
person[firstName]: Rajat
person[lastName]: Verma
person[country]: India
rajatverma@rajats-MacBook-Air _01_JavascriptBasics %
```

ii.


```

JS 28_JavascriptObject_ReAssign.js > ...
1 //Object -> It is a collection of properties
2
3 let person ={
4     firstName:'Rajat',
5     lastName:'Verma',
6     country:'India'
7 }
8
9 console.log("person.firstName: "+person.firstName)
10 console.log("person.lastName: "+person.lastName)
11 console.log("person.country: "+person.country)
12
13 person.firstName='Shreya'
14 person.lastName='Sharma'
15 person.country='India'
16
17 console.log("person.firstName: "+person.firstName)
18 console.log("person.lastName: "+person.lastName)
19 console.log("person.country: "+person.country)
20

```

iii.

```

JS 29_JavascriptObject_AddNewPropertyRuntime.js > ...
1 //Object -> It is a collection of properties
2
3 let person ={
4     firstName:'Rajat',
5     lastName:'Verma',
6     country:'India'
7 }
8
9 console.log("person.firstName: "+person.firstName)
10 console.log("person.lastName: "+person.lastName)
11 console.log("person.country: "+person.country)
12
13 person.gender='Male'
14 console.log("person.gender: "+person.gender)
15

```

iv.

v.

```
JS 30_JavascriptObject_DeletePropertyRuntime.js > ...
1 //Object -> It is a collection of properties
2
3 let person ={
4     firstName:'Rajat',
5     lastName:'Verma',
6     country:'India'
7 }
8
9 console.log("person.firstName: "+person.firstName)
10 console.log("person.lastName: "+person.lastName)
11 console.log("person.country: "+person.country)
12
13 //Adding the property gender into Javascript Object person
14 person.gender='Male'
15 console.log("person.gender: "+person.gender)
16
17 console.log("-----")
18
19 //Deleting the property gender from Javascript Object person
20 delete person.gender
21
22 console.log("person.firstName: "+person.firstName)
23 console.log("person.lastName: "+person.lastName)
24 console.log("person.country: "+person.country)
25 console.log("person.gender: "+person.gender)
```

vi.

```
JS 32_JavascriptObject_IterateOverObject.js > ...
1 //Object -> It is a collection of properties
2
3 let person ={
4     firstName:'Rajat',
5     lastName:'Verma',
6     country:'India'
7 }
8
9 for(let key in person){
10     console.log(key+" : "+person[key])
11 }
12
```

```
JS 33_JavascriptObject_PropertyHasFunction.js > ...
1 //Object -> It is a collection of properties
2
3 let person ={
4     firstName: 'Rajat',
5     lastName: 'Verma',
6     country: 'India',
7     fullName: function(){
8         return (this.firstName+" "+this.lastName)
9     }
10 }
11 console.log("person.fullName(): "+person.fullName())
12
```

vii.

p. Class and Object in same class

i. This concept is introduced from ES6 engine

```
JS 35_ClassAndObject_Constructors.js > Person
1 class Person{
2     firstName
3     lastName
4     country
5     age
6
7     getAllDetails(){
8         return (this.firstName + " " + this.lastName + " "+this.country+" "+this.age)
9     }
10
11     //Parameterized Constructor
12     constructor(firstName, lastName, country, age){
13         console.log("Parameterized Constructor called")
14         this.firstName = firstName;
15         this.lastName=lastName;
16         this.country=country;
17         this.age=age;
18     }
19 }
20
21 //Creating the object of class Person
22 let person_obj = new Person("Rajat", "Verma","India",27)
23 console.log("person_obj.firstName: "+person_obj.firstName)
24 console.log("person_obj.lastName: "+person_obj.lastName)
25 console.log("person_obj.country: "+person_obj.country)
26 console.log("person_obj.age: "+person_obj.age)
27
28 console.log("person_obj.getAllDetails(): "+person_obj.getAllDetails())
```

ii.



q. Class and Object in different class

```
JS 36_ClassPerson.js > <unknown> > Person > constructor
1 //class Person{
2 // This will make class Person as public and anyone can use this now
3 module.exports = class Person{
4     firstName
5     lastName
6     country
7     age
8
9     getAllDetails(){
10        return (this.firstName + " " + this.lastName + " "+this.country+" "+this.age)
11    }
12
13    //Parameterized Constructor
14    constructor(firstName, lastName, country, age){
15        console.log("Parameterized Constructor called")
16        this.firstName = firstName;
17        this.lastName=lastName;
18        this.country=country;
19        this.age=age;
20    }
21 }
```

i.

```
JS 36_ApplicationUsingClassPerson.js > ...
1 const Person = require('./36_ClassPerson')
2
3 //Creating the object of class Person
4 let person_obj = new Person("Rajat", "Verma","India",27)
5 // console.log("person_obj.firstName: "+person_obj.firstName)
6 // console.log("person_obj.lastName: "+person_obj.lastName)
7 // console.log("person_obj.country: "+person_obj.country)
8 // console.log("person_obj.age: "+person_obj.age)
9
10 console.log("person_obj.getAllDetails(): "+person_obj.getAllDetails())
11
12
13 let person_obj2 = new Person("Shreya", "Sharma","India",25)
14 console.log("person_obj2.getAllDetails(): "+person_obj2.getAllDetails())
```

ii.

r. OOPS

i. Inheritance

```
//Inheritance is the Main Pillar in Object oriented Programming
//one class can inherit/acquire the properties,Methods of another class
//The class which inherits the properties of other is known as subclass (derived class, child class)
//the class whose properties are inherited is known as superclass
```

1. To connect:

- a. <https://www.linkedin.com/in/rajat-v-3b0685128/>
- b. <https://github.com/rajatt95>
- c. <https://rajatt95.github.io/>

THANK YOU!

